

**JK Lakshmipat University**

*Institute of Engineering & Technology*

**LAB1 (ASSIGNMENT SET -1)**

*CS1134: Computer Organisation and Architecture*

**Submitted by**  
Name: Siddhi Nyati  
Roll No: 2022btech101

**Faculty Supervisor:**

Dr. Pranab Roy

Dr. Ajai Jain

Activity no: 1

1.2: Full Adder

Question 1.2: Design a VHDL model using dataflow architecture for full adder.

**Theory:**

A Full Adder is a combinational logic circuit that extends the functionality of a Half Adder by adding support for three binary inputs. It is composed of two XOR gates, two AND gates, and one OR gate, allowing it to add three single-bit binary digits: A, B, and C (the carry-in from a previous addition). The Full Adder produces two outputs: the 'Sum' and the 'Carry'.

**Boolean Expressions for Full Adder:**

* **Sum:** Sum=(A⊕B)⊕C
* **Carry**: Carry=(A⋅B)+(C⋅(A⊕B))

To derive the Sum for three inputs, we can express the output of the XOR gate: X=A⊕B⊕C=A′BC′+AB′C′+A′B′C+ABC=Sum of Full Adder

A diagram of a diagram of a diagram

Description automatically generated with medium confidence

A diagram of a circuit

Description automatically generated

**Truth Table for the Sum:**

|  |  |  |  |
| --- | --- | --- | --- |
| ***A (Input)*** | ***B (Input)*** | ***C (Input)*** | ***X (Output)*** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

For the Carry output, we can use the OR and AND gates:

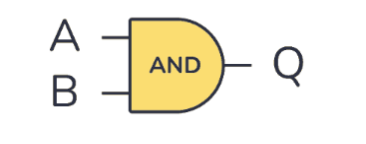
* **Boolean Expression for Carry: Q = A . B + C**

A yellow and black logo

Description automatically generated

**Truth Table for the OR Gate:**

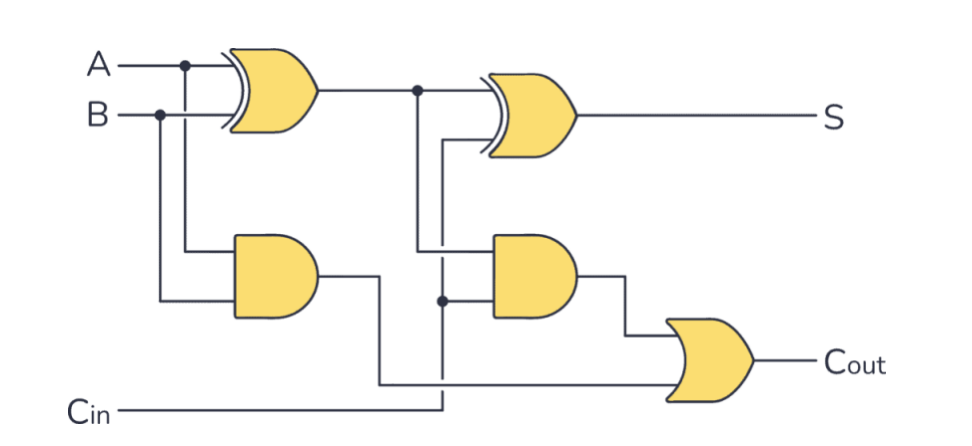
|  |  |  |
| --- | --- | --- |
| ***A (Input)*** | ***B (Input)*** | ***Q (Output) = A+B*** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

****

**Truth Table for the AND Gate:**

|  |  |  |
| --- | --- | --- |
| ***A (Input)*** | ***B (Input)*** | ***Q (Output) = A \cdot B*** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Combining the outputs from the two AND gates and one OR gate, we get the complete truth table for the Full Adder:

****

**Complete Truth Table for Full Adder:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***A (Input)*** | ***B (Input)*** | ***C (Input)*** | ***Sum (Output)*** | ***Carry (Output)*** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**Dataflow Modeling in VHDL**

VHDL, which stands for Very High-Speed Integrated Circuit Hardware Description Language, is an essential tool in digital circuit design. It allows for modeling and simulating hardware systems across various abstraction levels. VHDL supports three primary modeling styles: Dataflow, Behavioral, and Structural.

In this discussion, we focus on Dataflow modeling, which represents the flow of data between components without detailing their internal workings. This method uses logical operators like AND, OR, and NOT to describe the relationships among inputs and outputs.

The benefits of Dataflow modeling include its simplicity, efficiency, and clarity, making it an effective way to design digital systems such as the Full Adder.

**VHDL Code for Full Adder**

----------------------------------------------------------------------------------

-- Company:

-- Engineer:

--

-- Create Date: 12:45:50 09/23/2024

-- Design Name:

-- Module Name: full\_adder - Behavioral

-- Project Name:

-- Target Devices:

-- Tool versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

----------------------------------------------------------------------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity full\_adder is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

Cin : in STD\_LOGIC;

SUM : out STD\_LOGIC;

Cout : out STD\_LOGIC);

end full\_adder;

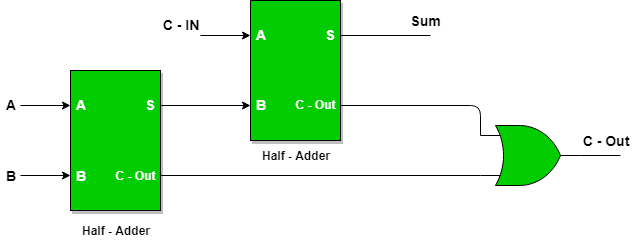
architecture dataflow of full\_adder is

begin

SUM <= A XOR B XOR Cin;

Cout <= (A AND B) OR (B AND Cin) OR (Cin AND A);

end dataflow;



**TEST BRANCH CODE FOR FULL ADDER**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity tb\_full\_adder is

end tb\_full\_adder;

architecture behavior of tb\_full\_adder is

-- Component declaration for the Unit Under Test (UUT)

component full\_adder

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

Cin : in STD\_LOGIC;

SUM : out STD\_LOGIC;

Cout : out STD\_LOGIC);

end component;

-- Signals to simulate inputs and outputs

signal A : STD\_LOGIC := '0';

signal B : STD\_LOGIC := '0';

signal Cin : STD\_LOGIC := '0';

signal SUM : STD\_LOGIC;

signal Cout : STD\_LOGIC;

begin

-- Instantiate the Unit Under Test (UUT)

UUT: full\_adder Port map (

A => A,

B => B,

Cin => Cin,

SUM => SUM,

Cout => Cout

);

-- Stimulus process

stim\_proc: process

begin

-- Test all combinations of inputs A, B, Cin

A <= '0'; B <= '0'; Cin <= '0'; wait for 10 ns;

A <= '0'; B <= '0'; Cin <= '1'; wait for 10 ns;

A <= '0'; B <= '1'; Cin <= '0'; wait for 10 ns;

A <= '0'; B <= '1'; Cin <= '1'; wait for 10 ns;

A <= '1'; B <= '0'; Cin <= '0'; wait for 10 ns;

A <= '1'; B <= '0'; Cin <= '1'; wait for 10 ns;

A <= '1'; B <= '1'; Cin <= '0'; wait for 10 ns;

A <= '1'; B <= '1'; Cin <= '1'; wait for 10 ns;

-- Finish simulation

wait;

end process;

end behavior;

